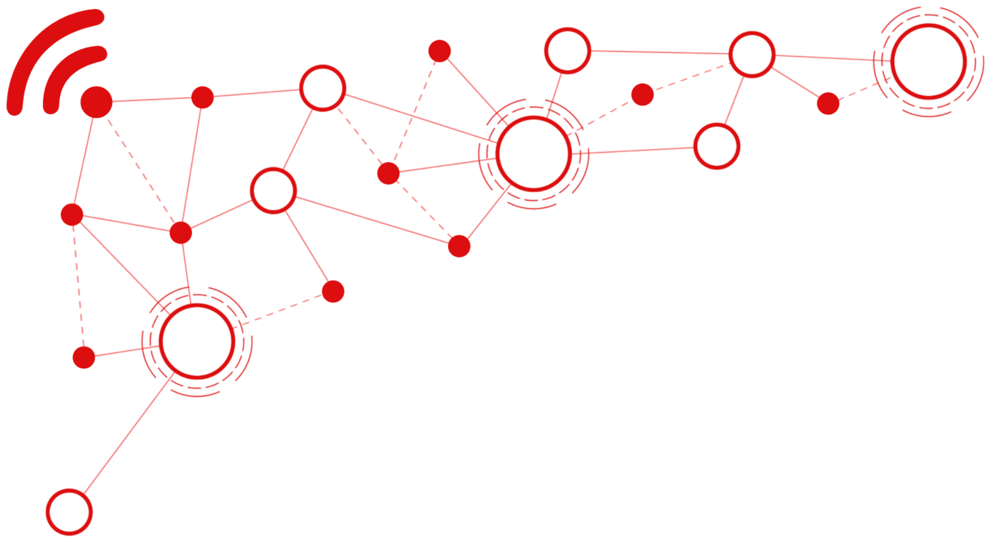
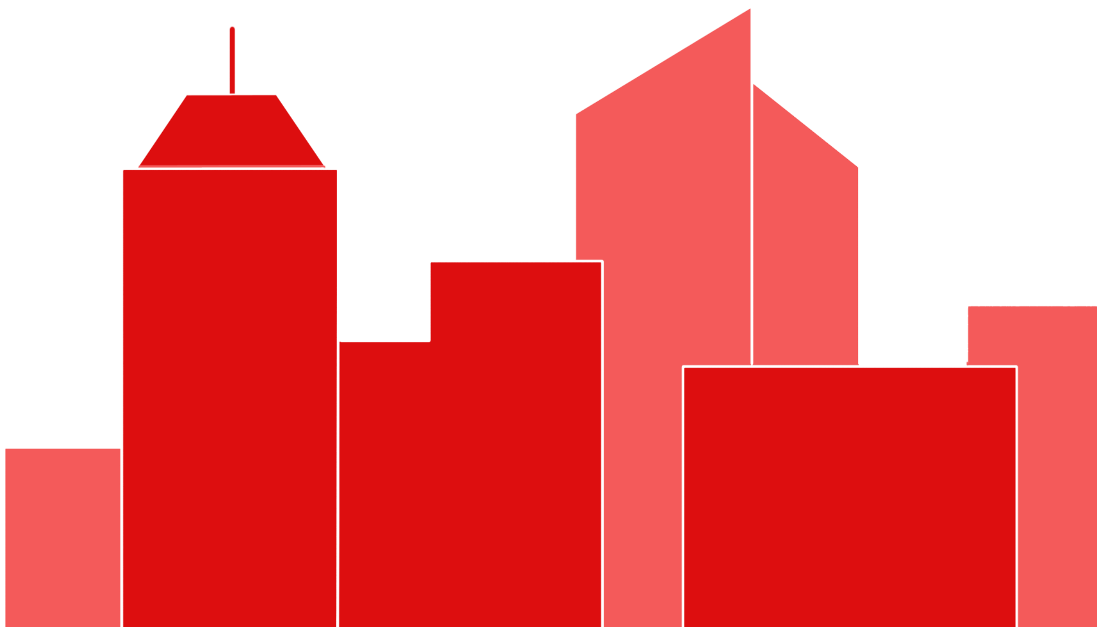


NOI.PH 2023

National Olympiad in Informatics

Finals Round 2



Contents

Notes

- Many problems have large input file sizes, so use fast I/O. For example:
  - In C/C++, use `scanf` and `printf`.
  - In Python, use `sys.stdin.readline()`
- On interactive problems, make sure to **flush** your output stream after printing.
  - In C++, use `fflush(stdout);` or `cout << endl;`
  - In Python, use `sys.stdout.flush()` or `print(flush=True)`
  - For more details, including for other languages, ask a question/clarification through CMS.
- Good luck and enjoy the problems!

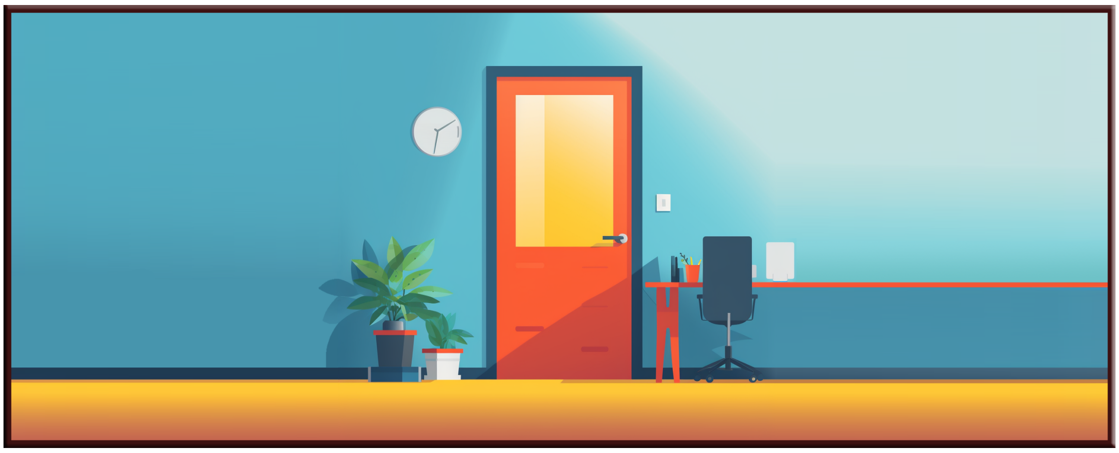
Important! Read the following:

- Your solution will be checked by running it against one or more (usually several) hidden test cases. You will not have access to these cases, but a correct solution is expected to handle them correctly.
- The output checker is **strict**. Follow these guidelines strictly:
  - It is **space sensitive**. Do not output extra leading or trailing spaces. Do not output extra blank lines unless explicitly stated.
  - It is **case sensitive**. So, for example, if the problem asks for the output in lowercase, follow it.
  - Do not print any tabs. (No tabs will be required in the output.)
  - Do not output anything else aside from what's asked for in the Output section. So, do not print things like "`Please enter t`".

  Not following the output format strictly and exactly will likely result in the verdict "*Output isn't correct*".
- Do not read from, or write to, a file. You must read from the standard input and write to the standard output.
- Only include one file when submitting: the source code (.cpp, .py, etc.) and nothing else.
- Only use letters, digits and underscores in your filename. Do not use spaces or other special symbols.
- Many problems have large input file sizes, so use fast I/O. For example:
  - In C/C++, use `scanf` and `printf`.
  - In Python, use `sys.stdin.readline()`

  We recommend learning and using these functions during the Practice Session.
- On interactive problems, make sure to **flush** your output stream after printing.
  - In C++, use `fflush(stdout);` or `cout << endl;`
  - In Python, use `sys.stdout.flush()` or `print(flush=True)`
  - For more details, including for other languages, ask a question/clarification through CMS.
- Good luck and enjoy the contest!

Problem A
Tulkinao



Supkotao[1] works a 9 to 5 job in an office. Usually, he plays harmless pranks in the office, like swapping the letters in signs to make new signs. But recently, he is being pranked by his coworker, Tulkiano, who is also swapping the letters in his signs.

Supkoato's signs are spelled-out numbers. Given a positive integer n, without any zero digits, he replaces each digit with its spelled-out Ilokano translation:

1. `maysa`
2. `dua`
3. `tallo`
4. `uppat`
5. `lima`
6. `innem`
7. `pito`
8. `walo`
9. `siam`

He takes the letters and uses it to make a sign, and then he goes home. After he goes home, Tilukano swaps the letters in his signs. He can add and remove any number of spaces, but he can't add or remove letters.

For example, if Supokato chose 957814, his sign would say `siam lima pito walo maysa uppat`. After he goes home, Tilaukno can swap the letters to say `siyam`

[1]`https://www.hackerrank.com/contests/noi-ph-practice-page/challenges/supkotao`
Warning: Don't open this URL during contest! Attempting to access the internet during the contest is grounds for disqualification.

`lima pito walo sampu apat`.

The next day, Sopuakto comes to the office and looks at the new sign. He wants to return it to the original, but he can't remember the n that he chose! Help him find a possible n, or say if there's no possible n.

As usual, ***tackle the subtasks first***.

Input Format

The first line of input contains s, the number of signs he sees. Then s descriptions of signs follow.

Each sign description consists of two lines. The first line contains an integer ℓ, the number of words in the sign. The second line contains the sign itself, which contains only lowercase letters and spaces.

Output Format

For each sign, output a single line containing either:

- a possible n for the sign, as described in the problem statement, or
- the string "`diak maawatan`" (without quotes) if no such n exists.

There may be multiple possible ns; any one will be accepted.

Constraints and Subtasks

For all subtasks
$1 \le s \le 420$ The sign consists only of lowercase English letters and spaces. There are no two consecutive spaces in the sign. There are no leading or trailing spaces in the sign.

Subtask	Points	Constraints
1	**30**	$\ell = 1$ Each word in the sign is at most 5 characters long.
2	**42**	$1 \le \ell \le 4$ Each word in the sign is at most 7 characters long.
3	**28**	$1 \le \ell \le 2023$ Each word in the sign is at most 9 characters long.

Sample I/O

Input

```
5
6
siyam lima pito walo sampu apat
4
poem mainit si payton
4
tula fetch si payton
9
ano dumi tao lipat luma tanim piso waley sampal
13
sa lupa may dusa sa dilim may saya at may sumpa tayo samasama
```

Output

```
957814
7167
diak maawatan
123456789
12342911911
```

Explanation

The first case is the example in the problem statement. Other possible answers would be 145789 and 987541.

In the second case, Supoakto's original sign could have said `pito maysa innem pito`, which Tilakuno could have changed to say `poem mainit si payton`.

In the third case, it is impossible for the sign `tula fetch si payton` to have been made by Tilaunko's swapping, as it contains an `f`, which doesn't exist in any Ilokano digits.

Problem B
Budol Sort

Last week, Cristiano Alberto decided to try his luck as a seller of *digital* products. His latest and greatest invention? Selling sequences of sorted *digits* on the internet!

Nobody ever thought anyone ever needed this. But that's true for half the things you can buy online these days. More importantly, rule 35 of the internet says: If it exists, *nasa Gazada 'yan*!

Cristiano has many sequences of unsorted digits. And now he has to sort them before he ships them off to his customers.

He bought six different sorting machines on the internet, but it turns out that none of them work as advertised. Some of them look like they'll work eventually, but some of them look like they'll do the opposite of what he wants! What's worse is that these machines look so cheap that they could break at any moment *and* they operate on microtransactions so each use costs him some **Xi'an coins** (西安币). Who knew that the Chinese city of Xi'an had their own separate currency?

These are the six machines he bought:

1. **Machine A** a.k.a. **the Backward+ Machine**:
   First reverses the sequence, and then shuffles to the next lexicographically larger permutation, or back to the smallest if there is no longer any larger arrangement. This machine can only be used at most u_A times, and consumes c_A Xi'an coins per use.

   For example, Machine A takes the sequence $[1, 3, 0, 4, 2, 5]$ to $[5, 2, 4, 1, 0, 3]$.

2. **Machine B** a.k.a. **the Backward− Machine**:
   First reverses the sequence, and then shuffles to the next lexicographically smaller permutation, or back to the largest if there is no longer any smaller arrangement. This machine can only be used at most u_B times, and consumes c_B Xi'an coins per use.

For example, Machine B takes the sequence $[1, 3, 0, 4, 2, 5]$ to $[5, 2, 4, 0, 1, 3]$.

3. **Machine C** a.k.a. **the Bubble+ Machine**:
Looks for the largest value not in its "proper position" in the sorted order[2] and swaps it with the element to its right. It does nothing to an already-sorted sequence. This machine can only be used at most u_C times, and consumes c_C Xi'an coins per use.

   For example, Machine C takes the sequence $[1, 3, 0, 4, 2, 5]$ to $[1, 3, 0, 2, 4, 5]$.

4. **Machine D** a.k.a. **the Bubble− Machine**:
Looks for the smallest value not in its "proper position" in the sorted order[3] and swaps it with the element to its left. It does nothing to an already-sorted sequence. This machine can only be used at most u_D times, and consumes c_D Xi'an coins per use.

   For example, Machine D takes the sequence $[1, 3, 0, 4, 2, 5]$ to $[1, 0, 3, 4, 2, 5]$.

5. **Machine E** a.k.a. **the Weave+ Machine**:
First *Faro outshuffles* the sequence, and then shuffles to the next lexicographically larger permutation, or back to the smallest if there is no longer any larger arrangement. This machine can only be used at most u_E times, and consumes c_E Xi'an coins per use.

   For example, Machine E takes the sequence $[1, 3, 0, 4, 2, 5]$ to $[1, 4, 3, 2, 5, 0]$.

6. **Machine F** a.k.a. **the Weave− Machine**:
First *Faro outshuffles* the sequence, and then shuffles to the next lexicographically smaller permutation, or back to the largest if there is no longer any smaller arrangement. This machine can only be used at most u_F times, and consumes c_F Xi'an coins per use.

   For example, Machine F takes the sequence $[1, 3, 0, 4, 2, 5]$ to $[1, 4, 3, 0, 5, 2]$.

The **Faro outshuffle**, also known as the **weave outshuffle**, works by splitting the sequence into two equal parts—the left and the right—and interleaving them perfectly, with the leftmost digit of the left half ending up as the leftmost digit. For example, the Faro outshuffle takes $[0, 1, 2, 3, 4, 5]$ to $[0, 3, 1, 4, 2, 5]$. If the sequence has an odd length, then the split is done such that the length of the left part is one more than the length of the right part. For example, the Faro outshuffle takes $[0, 1, 2, 3, 4]$ to $[0, 3, 1, 4, 2]$.

Cristiano doesn't have any time to wait for these machines to run their course, so he decides to improvise and use these machines *in alphabetical order*. In other words, *once a machine is used, the machines before it cannot be used anymore*. For example, once Machine C is used, Machines A and B cannot be used anymore.

[2] in other words, the largest value whose right-side neighbor is smaller than itself
[3] in other words, the smallest value whose left-side neighbor is larger than itself

Cristiano is only interested in selling sequences of length n that are permutations of the digits $0, 1, 2, \ldots, n-1$. We'll call these sequences **complete**. Note that $n < 10$ because an average human has less than 10 digits in both hands.

Cristiano initially has s complete sequences at his disposal. He doesn't have any other sequences. And because he's in a rush, he considers a sequence **budol-sorted** if it belongs to a given collection with t complete sequences. (These t sequences don't necessarily look sorted, but Cristiano's okay with it since he knows buyers aren't picky.)

Determine whether or not it's possible to get a valid budol-sorted sequence using the machines, starting from any one of the s complete sequences he initially has. If it's possible, determine the minimum cost needed to get a valid budol-sorted sequence.

Input Format

The first line of input contains three space-separated integers n, s and t.

The next s lines describe the s complete sequences Cristiano initially has. The ith of these lines contains n space-separated integers between 0 and $n-1$ denoting the elements of the ith such complete sequence.

The next t lines describe the t complete sequences Cristiano considers budol-sorted. The ith of these lines contains n space-separated integers between 0 and $n-1$ denoting the elements of the ith such complete sequence.

The next line contains 6 space-separated integers $c_A, c_B, c_C, c_D, c_E, c_F$.

The next line contains 6 space-separated integers $u_A, u_B, u_C, u_D, u_E, u_F$. This is the final line.

Output Format

Output a line containing an integer, the minimum cost needed to get a valid budol-sorted sequence, or the string "`BUDOL`" (without the quotes) if it is impossible to get a valid budol-sorted sequence.

Constraints and Subtasks

For all subtasks
$5 \leq n < 10$ $1 \leq s, t \leq 8$ $1 \leq c_A, c_B, c_C, c_D, c_E, c_F \leq 3 \cdot 10^9$ $0 \leq u_A, u_B, u_C, u_D, u_E, u_F \leq 3 \cdot 10^9$

Subtask	Points	Constraints
1	**16**	$n \le 6$ $u_A, u_B, u_C, u_D, u_E, u_F \le 5$
2	**13**	$n \le 6$
3	**9**	$n \le 7$
4	**12**	$n \le 8$
5	**14**	$u_A = u_F = 0$
6	**14**	$u_A = u_B = u_C = u_D = u_E = u_F = 3 \cdot 10^9$
7	**22**	No additional constraints.

Sample I/O

Input 1

```
5 2 3
3 2 4 1 0
0 1 4 3 2
2 1 4 0 3
1 3 2 4 0
2 1 3 0 4
42 69 32 111 36 127
3 3 3 1 1 2
```

Output 1

```
169
```

Input 2

```
5 5 1
1 0 2 3 4
0 2 1 3 4
0 1 3 2 4
0 1 2 4 3
4 3 2 1 0
0 1 2 3 4
2023 2023 2023 2023 2023 2023
0 0 0 0 0 0
```

Output 2

```
BUDOL
```

Explanation

For the first sample input, here's one way to get a budol-sorted sequence:

- Start with the sequence $[0, 1, 4, 3, 2]$.
- Use Machine B to get $[2, 3, 4, 0, 1]$.
- Use Machine C to get $[2, 3, 0, 4, 1]$.
- Use Machine C to get $[2, 3, 0, 1, 4]$.
- Use Machine E to get $[2, 1, 4, 0, 3]$.

The sequence $[2, 1, 4, 0, 3]$ is now budol-sorted, and the total cost is $69 + 32 + 32 + 36 = 169$ Xi'an coins, which can be shown to be the minimum total cost possible to get a budol-sorted sequence.

In the second sample input, $u_A = u_B = u_C = u_D = u_E = u_F = 0$. In other words, Cristiano cannot use any of the machines at all, so he cannot produce the budol-sorted sequence $[0, 1, 2, 3, 4]$. He got scammed!

Problem C
Maglalatik Dance-Off

Three years after the first Budots Dance-Off, a spin-off competition called The First Annual Maglalatik Dance-Off Competition was born. This competition involves n dancers as competitors. In this competition, each pair of dancers have one dance-off together. For each pair, the judges name **exactly one** winner.

Aling Divina Wata believes that dancing has a very big cosmic influence in the world. This is how she discovered the First Annual Maglalatik Dance-Off Competition to begin with. After becoming a world-renowned essential oils enthusiast and a top-level multilevel marketer, she wanted to find a new source of passive income using her mindset. She decided to start betting in Dance-Off Competitions.

D. Wata ranks the contestants of the competition from 1 to n where 1 is (who she thinks is) the strongest dancer and n is who she thinks is the weakest. D. Wata's predictions were correct, with the exception of k so-called **underdog** matches. In an *underdog* match, the dancer which she ranked lower (i.e., larger number) won.

Each of the $\frac{n(n-1)}{2}$ dance-offs are captured on video for archival purposes. In each archival dance-off video, only the two dancers in the dance-off are visible.

D. Wata requested you to make a compilation video using the archival footages so that she can watch it later on to study the dancers and make better predictions for the Second First Annual Maglalatik Dance-Off Competition. She has a few constraints, however:

- The compilation video needs to feature exactly $n-1$ different dance-offs in their entirety because $n-1$ is her lucky number![4]
- Each of the n dancers must appear in the video compilation.

[4]Yes, her lucky number is dependent on the number of dancers in an obscure dance competition.

- For $i = 1, \ldots, n-2$, the winner of the ith dance-off in the compilation video must be the loser in the $(i+1)$th dance-off featured in the compilation video.

How many different compilation videos can you make given these constraints? Since the answer can be very large, only output it modulo 998244353, a prime number.

Input Format

The first line of input contains two space-separated integers n and k, the number of dancers and the number of *underdog* matches, respectively.

The next k lines describe the *underdog* matches. The ith of these lines contains two space-separated integers w_i and ℓ_i, indicating that dancer ranked w_i won over the dancer ranked ℓ_i.

Output Format

Output a single line containing an integer denoting the number of different compilation videos modulo 998244353.

Constraints and Subtasks

For all subtasks
$1 \leq n \leq 1000$
$1 \leq \ell_i < w_i \leq n$
$0 \leq k \leq n(n-1)/2$
All (w_i, ℓ_i) pairs are distinct.

Subtask	Points	Constraints
1	**25**	$k \leq 8$, $n \leq 8$
2	**24**	$k \leq 8$, $n \leq 16$
3	**20**	$k \leq 8$
4	**16**	$k \leq 10$
5	**9**	$k \leq 12$
6	**6**	$k \leq 14$

Sample I/O

Input	Output

5 3
3 1
5 4
4 2
	9

Explanation

We refer to the dancer that D. Wata ranked i as dancer i.

In this example, there were 5 dancers, and 3 *underdog* matches:

- Dancer 3 won in their dance-off against dancer 1.
- Dancer 5 won in their dance-off against dancer 4.
- Dancer 4 won in their dance-off against dancer 2.

That means D. Wata correctly predicted 7 matches:

- Dancer 1 won in their dance-off against dancer 2.
- Dancer 1 won in their dance-off against dancer 4.
- Dancer 1 won in their dance-off against dancer 5.
- Dancer 2 won in their dance-off against dancer 3.
- Dancer 2 won in their dance-off against dancer 5.
- Dancer 3 won in their dance-off against dancer 4.
- Dancer 3 won in their dance-off against dancer 5.

An example compilation video would be:

1. the dance-off between dancers 5 and 4, then
2. the dance-off between dancers 3 and 5, then
3. the dance-off between dancers 2 and 3, then
4. the dance-off between dancers 1 and 2.

In this video, the winner of the first dance-off is 5, who is then loses to 3 in the second dance-off, who then loses to 2 in the third dance-off, who then loses to 1 in the last dance-off. It can be checked that there are 9 possible compilation videos satisfying the constraints.

Problem D
I Know What You Did Last Summer

It's been many years since Camila was a little lovestruck señorita desperate for Shawn's affection. Funny thing, she actually did end up getting together with him, only she had to confront the reality that he was a little more toxic than she was expecting.

- *First of all, he was way too clingy. He couldn't have one drink without telling her that he's thinking about her. And when she asked for space, he would pout and say that* everything means nothing if he can't have her.
- *Whenever she called him out on his unhealthy behavior, he would play victim and ask her to* please have mercy on him.
- *He was controlling, too, always demanding that she keep him updated on everywhere she goes. And he would get confrontational and accuse her of lying with just the slightest provocation.*

Camila had eventually realized she deserved better, and that it was time for her to work on herself and her own identity first. So when Shawn demanded she look into his eyes and tell him what she did last summer, *she gave him an honest answer:* She joined NOI.PH's IOI Training Camp and grinded until she was a world-class competitive programmer! That's why she was too busy for him!

Now one of the top competitive programmers in the Philippines, Camila finds herself faced with n *fans* who desperately want to meet her, labeled from 1 to n in the order that they should get their autograph (1 first and n last). The fans are to be arranged into two *queues*, and we note that the **initial configuration** of fans can be completely described by which queue each fan is in, and in what order.

Camila has to sign the n autographs in order, and only has the following two operations available.

- Send the fan from the **front** of either queue to the **back** of the other queue. This costs 1 unit of her energy.
- Give an autograph to the **front** person of either queue. This costs 0 units of her energy.
    - Again, she can only give an autograph to person j if she has already given an autograph to all persons i such that $1 \leq i < j$ (always true for person 1).
    - Note that the fan **does not leave the queue**.

Let's say that the **potential** of an initial configuration is equal to the minimum amount of energy needed to give all autographs to all n fans in the proper order. In symbols, if C is some initial configuration, then let this minimum amount of energy be denoted by potential(C).

Now, Camila wouldn't be a world-class competitive programmer if she didn't always have some wacky problem in her mind. Let's suppose that she has some positive integer k in mind. She's interested in two types of questions.

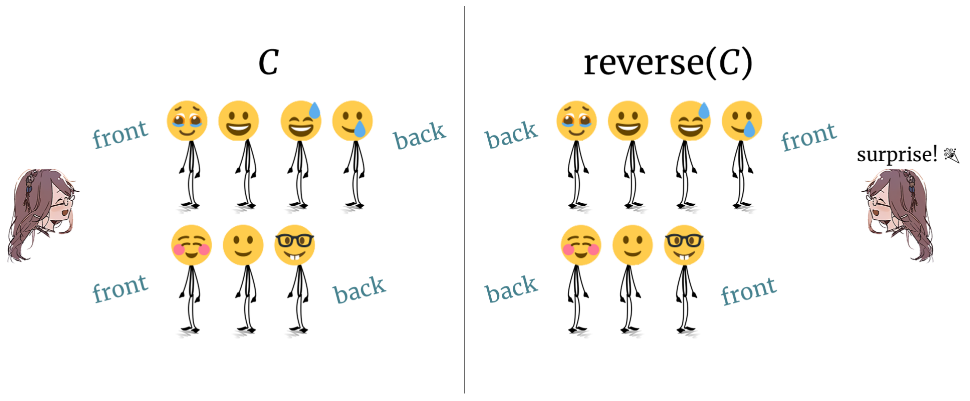
- **Type 1:** Given n and k, find any initial configuration C such that

$$\text{potential}(C) = k.$$

- **Type 2:** Given n and k, find any initial configuration C such that

$$\text{potential}(C) - \text{potential}(\text{reverse}(C)) = k.$$

    - Here, reverse(C) is the configuration you get by swapping which end is the front and back of each queue, but keeping the positions of each fan within each queue fixed in place. Perhaps Camila surprises her fans by entering from the opposite direction! See the following diagram.



Input Format

The first line of input contains a single integer s, which is either 1 or 2, denoting the question type. This signifies that all test cases in the input will be of Type s.

The second line of input contains t, the number of test cases. The descriptions of t test cases follow.

Each test case consists of a single line containing two space-separated integers n and k.

Output Format

For each test case, output a single line containing the string (without the quotes)

- "`I KNOW`" if there is an initial configuration C such that $\text{potential}(C) = k$ if $s = 1$, or $\text{potential}(C) - \text{potential}(\text{reverse}(C)) = k$ if $s = 2$, or
- "`I DUNNO`" otherwise.

In addition, if there exists such an initial configuration, output three more lines describing one such configuration.

- The first of these lines must contain two space-separated nonnegative integers ℓ_1 and ℓ_2, the number of people in the first and the second queues, respectively.
- The second of these lines must contain ℓ_1 space-separated positive integers denoting the labels of the fans in the first queue from front to back.
- The third of these lines must contain ℓ_2 space-separated positive integers denoting the labels of the fans in the second queue from front to back.

Note: If you outputted $\ell_1 = 0$ or $\ell_2 = 0$, then the corresponding line must be empty, but must still exist.

If there are multiple possible answers, any one will be accepted.

Constraints and Subtasks

For all subtasks
$1 \leq t \leq 5000$
$n \geq 1$
The sum of the ns in a single file is $\leq 10^6$.
$\lvert k\rvert \leq 10^9$
$s \in \{1, 2\}$

Subtask	Points	Constraints
1	**18**	$s = 1$, $n \le 8$
2	**17**	$s = 1$, $n \le 30$
3	**20**	$s = 1$, $n \le 300$
4	**5**	$s = 1$, $n \le 10000$
5	**1**	$s = 2$, $n \le 8$
6	**10**	$s = 2$, $n \le 30$
7	**23**	$s = 2$, $n \le 300$
8	**6**	$s = 2$, $n \le 10000$

Sample I/O

Input 1

```
1
3
5 7
5 9
8 -8
```

Output 1

```
I KNOW
3 2
1 4 2
3 5
I KNOW
3 2
2 4 1
5 3
I DUNNO
```

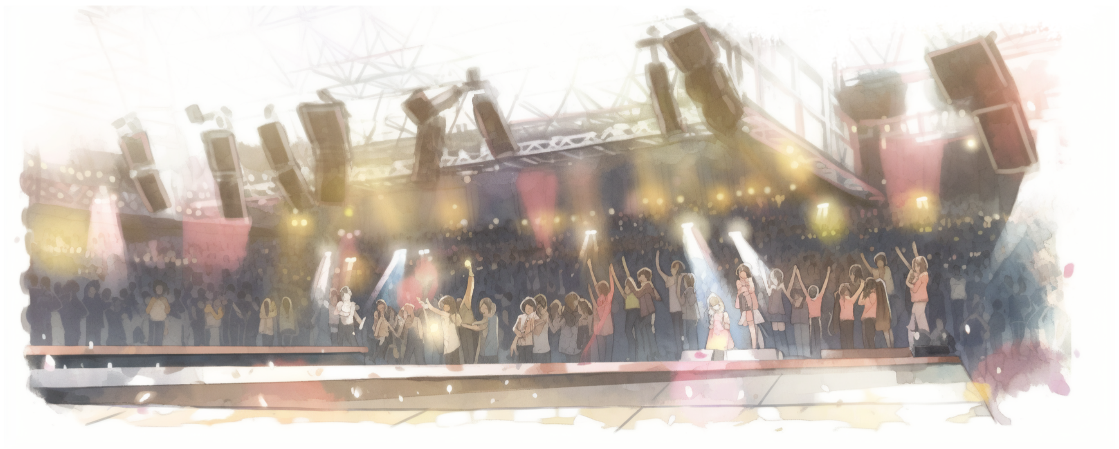
Input 2

```
2
1
5 -2
```

Output 2

```
I KNOW
3 2
1 4 2
3 5
```

Problem E
Ang Talon ni Maria Cristina



Maria Cristina is a famous rock singer in Iligan known for doing covers of Filipino songs such as Buwan by Juan Karlos, Wag Ka Nang Umiyak by Sugarfree, and Mula sa Puso by Jude Michael.

During her live concerts, she is known for doing linear mosh pits. This involves her jumping off the stage and then being caught and passed around by her fans.

More specifically, n of her fans form a straight line in front of the stage and we index these fans from 1 to n.

Her manager chooses two integers a and b such that $1 \le a \le b \le n$. Maria Cristina then chooses an integer i between a and b inclusive and jumps to fan i.

After being caught by fan i, she jumps to any fan *adjacent* to fan i as long as they exist and their index is at least a and at most b. She gets caught by that new fan, jumps to another adjacent fan whose index is at least a and at most b, and so on.

Fans a to b have ordered VIP passes though! Because of this, Maria Cristina is contractually obliged to let the ith fan catch her **exactly** x_i times for some integer x_i, for each i such that $a \le i \le b$.

Will she be able to do her signature mosh pit exactly once and fulfill all her contractual obligations for the fans numbered a through b inclusive?

If it's not possible to do so, output the minimum number of *changes* that must be made to the x_i values such that she can fulfill her contractual obligations after exactly one mosh pit. One **change** is defined as choosing an i between 1 and n inclusive and then either adding 1 to x_i or subtracting 1 from x_i. An x_i can be changed multiple times.

Note that you have to answer q questions of this kind, each with a different pair (a, b). Also, the questions are **independent** of each other, that is, the changes to be done are not actually applied after that question.

Input Format

The first line of input contains two space-separated integers n and q, whose meanings are described in the problem statement.

The second line contains n space-separated integers $x_1, x_2, \ldots, x_n$.

The next q lines describe the q questions. The ith of these lines contains two space-separated integers a and b denoting the (a, b) pair for the ith question.

Output Format

For each question, output a single line containing either:

- the string "`DAOG`" (without the quotes) if it is possible to fulfill Maria Cristina's contractual obligations to fans a through b, otherwise:
- an integer c denoting the minimum number of *changes* that must be made to the x_i so that Maria Cristina could fulfill her contractual obligations to fans a through b.

Constraints and Subtasks

For all subtasks
$1 \le n \le 360000$
$1 \le q \le 256000$
$q \le n(n+1)/2$
$\lvert x_i \rvert \le 5 \cdot 10^7$
$1 \le a \le b \le n$
No two questions have the same (a, b) pair.

Subtask	Points	Constraints
1	**15**	$n \le 32$
2	**7**	$n \le 256$
3	**10**	$n \le 640$
4	**11**	$n \le 6400$
5	**20**	$n \le 128000$, $q \le 160$
6	**37**	No additional constraints.

Sample I/O

Input	Output

```
17 5
2 3 3 2 1 1 2 1 0 1 2 5 2 1 -3 -3 -3
1 9
1 3
10 14
15 16
16 17
```
	```
	DAOG
	1
	2
	7
	7
	```

Explanation

In the first question, $(a, b) = (1, 9)$. Maria Cristina can fulfill her contractual obligations without changing any of the x_is; for example, she can initially jump on fan $i = 1$ and then continue via the following sequence of jumps:

$$1 \to 2 \to 3 \to 4 \to 3 \to 2 \to 1 \to 2 \to 3 \to 4 \to 5 \to 6 \to 7 \to 8 \to 7.$$

Thus, the output is `DAOG`.

In the second question, $(a, b) = (1, 3)$. In this case, Maria Cristina cannot fulfill her contractual obligations, but she can do so after just one *change*. For example, $x_3 = 3$ can be changed to $x_3 = 2$. After this change, she can now fulfill her obligation; for example, she can initially jump on fan $i = 1$ and then continue via the following sequence of jumps:

$$1 \to 2 \to 3 \to 2 \to 1 \to 2 \to 3.$$

Since one change is needed, the output is 1.

Alternatively, $x_2 = 3$ instead can be changed to $x_2 = 4$. After this change, she can now fulfill her obligation; for example, she can initially jump on fan $i = 3$ and then continue via the following sequence of jumps:

$$3 \to 2 \to 1 \to 2 \to 3 \to 2 \to 1 \to 2 \to 3.$$

In the third question, $(a, b) = (10, 14)$. Two changes are needed to fulfill her obligations. For example, $x_{12} = 5$ can be changed to $x_{12} = 4$ and then to $x_{12} = 3$. She can now fulfill her obligation; for example, she can initially jump on fan $i = 12$ and then continue via the following sequence of jumps:

$$12 \to 11 \to 10 \to 11 \to 12 \to 13 \to 14 \to 13 \to 12.$$

Since two changes are needed, the output is 2.

The fourth and fifth questions have the same answer, 7. Since all the x_i in the range in these questions are negative, we are forced to make them all at least 0 (since jumping on someone a negative number of times is nonsensical), with at least one of them needing to be at least 1 (the one Maria Cristina first jumps to). This also illustrates that the queries are independent, since we didn't actually apply these 7 changes after answering the questions.